Introducción a las plantillas en Catalyst Center

Introducción

Este documento describe Cisco Catalyst Center y la experiencia con plantillas de configuración para arquitecturas de campus de núcleo concentrado o de tres niveles.

Antecedentes

Este documento está dirigido a profesionales de la empresa que tengan conocimientos básicos sobre Cisco Catalyst Center y experiencia con las plantillas de configuración. Es especialmente relevante para aquellos que han trabajado o piensan trabajar con arquitecturas de campus de núcleo concentrado o de tres niveles.

El objetivo principal es ayudar a los lectores a implementar y automatizar las soluciones de configuración y gestión mediante plantillas en Cisco Catalyst Center. Mediante la presentación de perspectivas avanzadas, técnicas prácticas y ejemplos reales, este documento sirve como un recurso práctico para aquellos que buscan mejorar sus habilidades de infraestructura LAN y optimizar los flujos de trabajo a través de la automatización y la gestión basada en plantillas.

Resumen ejecutivo

A medida que las redes empresariales continúan evolucionando, la necesidad de una gestión escalable, coherente y automatizada nunca había sido tan acuciante. Cisco Catalyst Center proporciona una plataforma centralizada y basada en objetivos que simplifica la configuración, el aprovisionamiento y la garantía en las redes de las instalaciones. Este informe técnico analiza cómo los profesionales de la red pueden aprovechar las funciones de automatización y el editor de plantillas CLI de Cisco Catalyst Center para optimizar las operaciones de red, reducir los errores de configuración y acelerar las implementaciones en arquitecturas de núcleo de tres niveles y contraídas. Detalla las prácticas recomendadas para diseñar plantillas modulares basadas en Jinja2, integrar la automatización en flujos de trabajo de día 0 y día N, y lograr la coherencia operativa en las capas de núcleo, distribución y acceso. Al adoptar las estrategias descritas en este documento, puede transformar la gestión manual de redes tradicional en un modelo ágil, estandarizado y basado en la automatización alineado con la visión de red basada en objetivos de Cisco.

Retos de las redes de campus

A medida que las redes de campus evolucionan para satisfacer las demandas de las organizaciones modernas, se enfrentan a varios retos clave:

2a. Complejidad en la gestión de redes

Muchas funciones de red se siguen gestionando manualmente, lo que aumenta el riesgo de errores humanos. Esto no solo aumenta los esfuerzos de mantenimiento, sino que también supone una carga para los recursos de TI, especialmente con presupuestos estáticos o limitados.

2 ter. Retos de implementación y automatización

La incorporación de nuevos dispositivos para redes por cable e inalámbricas suele llevar mucho tiempo y ser compleja, lo que provoca retrasos en la implementación y un aumento de los costes administrativos.

2 quáter. Gestión de imágenes de software

Mantener una "imagen de oro" uniforme en toda la red es un reto. Muchas redes acaban con varios sistemas operativos para dispositivos por cable e inalámbricos, lo que provoca ineficiencias y dificultades de gestión.

2d. Configuraciones de red incoherentes

Las variaciones en la configuración de la red pueden dar lugar a problemas de conformidad e ineficiencias operativas, lo que dificulta el mantenimiento de una red fiable y segura.
2 sexies Expectativas de los usuarios en aumento

Los usuarios demandan una conectividad ininterrumpida y experiencias de aplicación sin problemas, independientemente de su ubicación o dispositivo. Para cumplir estas expectativas, las redes deben ser resistentes, inteligentes y capaces de adaptarse a los cambios en tiempo real.

Además de estos retos, las infraestructuras LAN modernas se enfrentan a otras muchas complejidades.

Simplificación de las redes de instalaciones con Cisco Catalyst Center

Cisco Catalyst Center es una solución de gestión de red centralizada para redes de campus que admite sedes centrales, sucursales, conexiones por cable e inalámbricas y entornos de TI/TO. Ofrece opciones de implementación flexibles, incluidos dispositivos físicos, servidores VMware ESXi o nube AWS. Con sus completas funciones, Catalyst Center simplifica las operaciones, mejora el rendimiento y refuerza la seguridad.

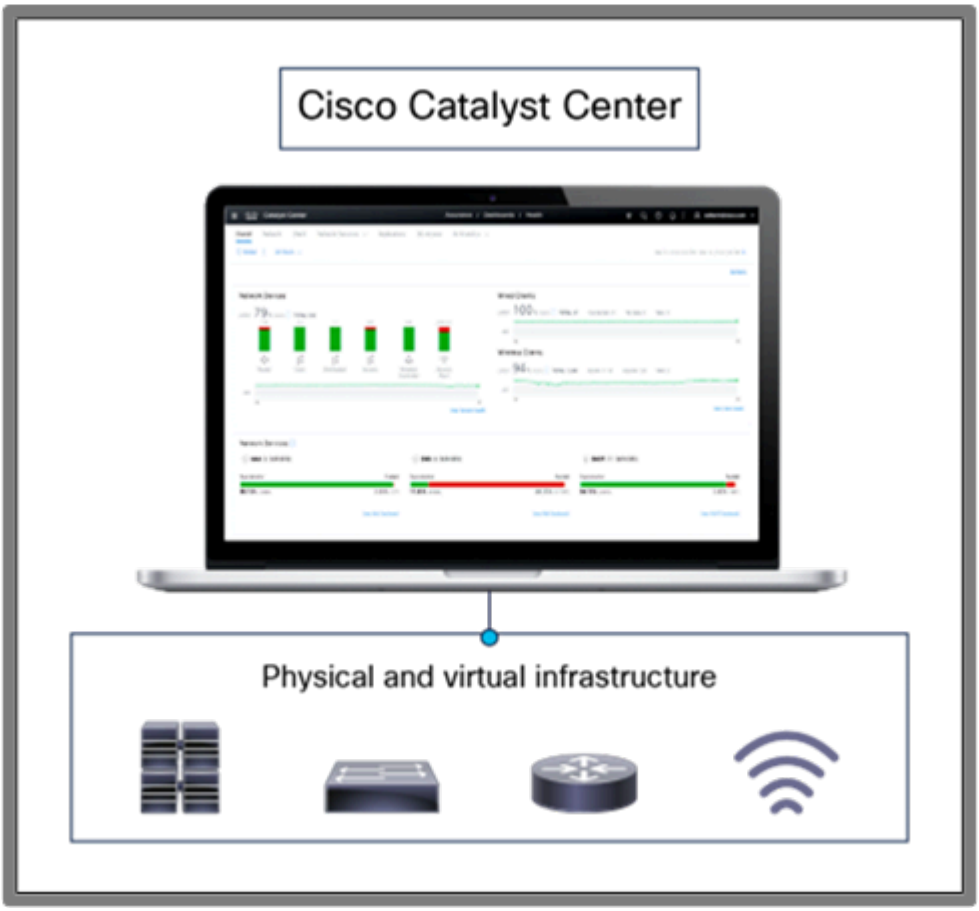


Figura 1: Administración de la infraestructura con Cisco Catalyst Center

Características y ventajas clave

Cisco Catalyst Center (CC) proporciona funciones avanzadas que optimizan la gestión y la automatización de la red:

- Aprovisionamiento sin intervención (ZTP): Automatiza la incorporación de dispositivos, lo que reduce el esfuerzo manual y el tiempo de implementación.

- Administración de imágenes de software (SWIM): Garantiza la coherencia de las versiones de software en todos los dispositivos con comprobaciones previas y posteriores a la actualización para evitar problemas.

- Automatización basada en intención: Simplifica las implementaciones convirtiendo la

intención de red en configuraciones de dispositivos para redes por cable e inalámbricas.

Automatización de LAN: Automatiza el direccionamiento y el routing IP de capa 3 para crear topologías de extremo a extremo.

Automatización de la red inalámbrica: Funciones como Plug and Play (PnP) permiten un aprovisionamiento rápido de puntos de acceso inalámbricos.

Gestión jerárquica de la red: Permite perfiles específicos del sitio (por ejemplo, SSID, parámetros de radiofrecuencia o VLAN) para realizar implementaciones uniformes en todas las ubicaciones.

Plantillas CLI: Catalyst Center Template Editor permite a los administradores crear y administrar fácilmente plantillas de configuración basadas en CLI, lo que permite una implementación coherente y eficaz en todos los dispositivos.

Garantía: Assurance permite tener una supervisión centralizada de los dispositivos administrados a través de CC.

Además de estas funciones, Cisco Catalyst Center ofrece muchas más funciones que están fuera del alcance de este documento. Este documento se centra principalmente en el diseño de plantillas CLI mediante Catalyst Center.

Descripción general de alto nivel de la arquitectura de instalaciones LAN con Catalyst Center

Las redes LAN tradicionales de campus forman la columna vertebral de la conectividad empresarial, lo que garantiza una comunicación fiable y escalable para dispositivos por cable e inalámbricos. Estas redes se suelen diseñar utilizando la arquitectura de 3 niveles o la arquitectura de núcleo contraído, en función del tamaño y la complejidad de la organización.

Arquitectura de tres niveles

La arquitectura de tres niveles es un modelo de diseño de red básico que consta de la capa principal, la capa de distribución y la capa de acceso. Esta arquitectura ofrece escalabilidad, alto rendimiento y gestión eficaz del tráfico. Consulte la descripción general de cada capa.

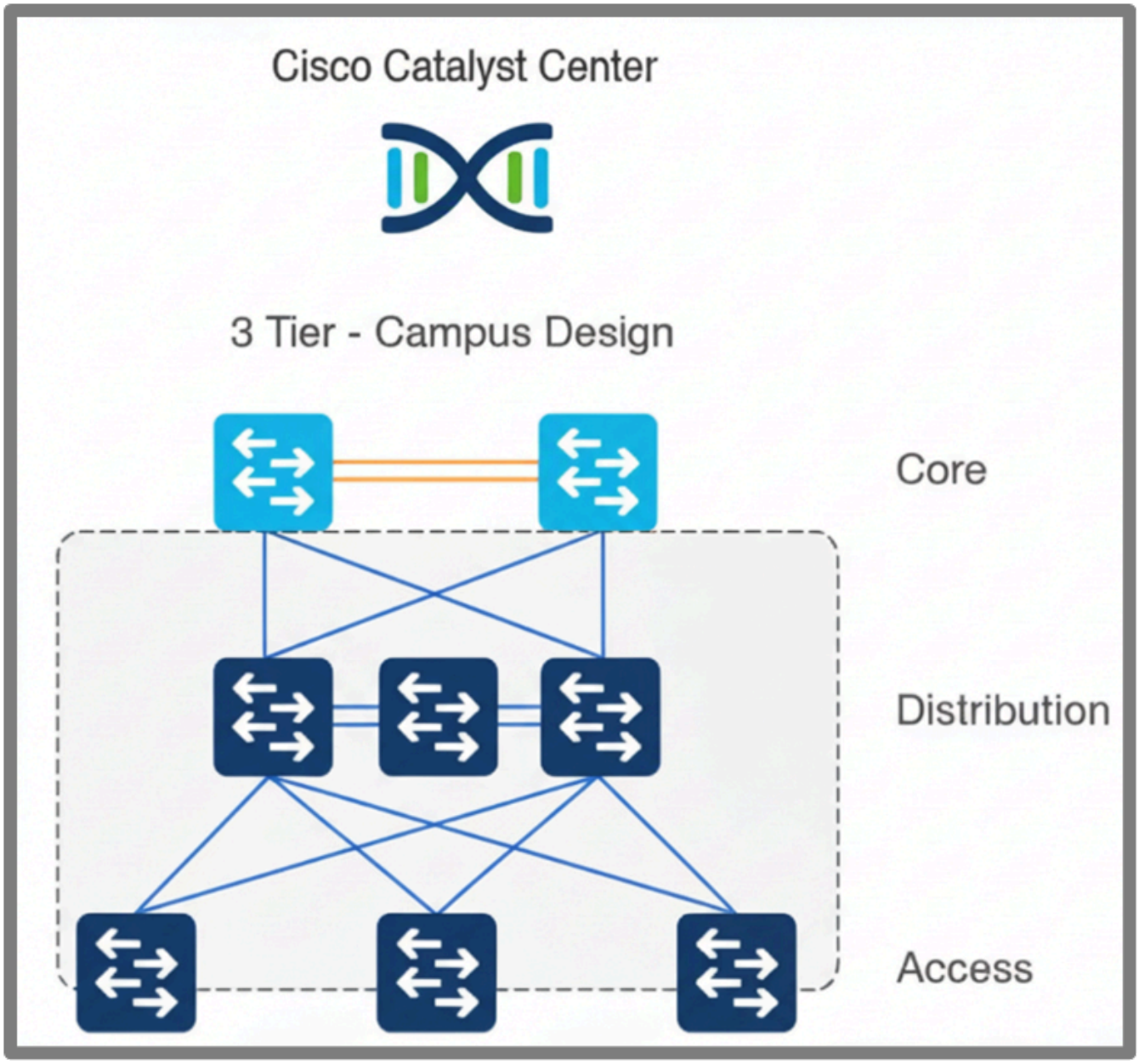


Figura 2: Arquitectura de campus de tres niveles

Capa de núcleo

La capa de núcleo actúa como la red troncal, proporcionando una conectividad y una escalabilidad de alta velocidad. Las configuraciones clave incluyen protocolos de ruteo ascendentes y descendentes (como OSPF y BGP), políticas de ruta, configuraciones de interfaz de enlace descendente y enlace ascendente, refuerzo de la seguridad, etc

Capa de distribución

La capa de distribución une las capas de núcleo y acceso, gestionando la agregación de tráfico, la aplicación de políticas y la redundancia. Las configuraciones clave incluyen HSRP/VRRP para redundancia, STP para prevención de bucles, VLAN de capa 2 y capa 3,

configuraciones de interfaz de enlace ascendente y descendente, ACL para seguridad y refuerzo de la seguridad.

Capa de acceso

La capa de acceso conecta los terminales a la red, lo que permite un acceso seguro y fiable. Las configuraciones clave incluyen la configuración de la interfaz de acceso, la configuración de la interfaz de enlace ascendente, VLAN de capa 2, ACL para restringir el acceso al dispositivo y refuerzo de la seguridad.

Arquitectura de núcleo contraída

La arquitectura de núcleo concentrado combina las capas de núcleo y distribución en una única capa, lo que reduce la complejidad y los costes a la vez que mantiene el rendimiento y la escalabilidad. Este enfoque es idóneo para redes pequeñas y medianas en las que no se requiere una capa de núcleo independiente. Consulte la descripción general de las capas de esta arquitectura.

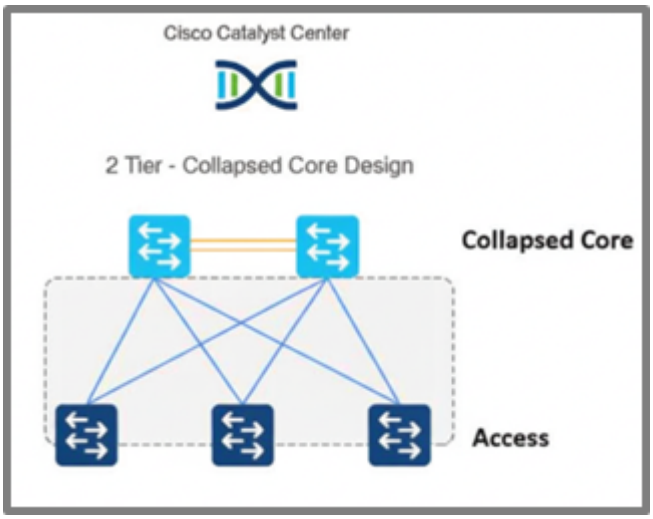


Figura 3: Arquitectura de campus central contraída

Capa de núcleo contraída

La capa de núcleo concentrado combina las funciones de las capas de núcleo y distribución, lo que proporciona conectividad de red troncal, agregación de tráfico y aplicación de políticas. Las configuraciones clave incluyen protocolos de ruteo ascendentes y descendentes (como OSPF y BGP), políticas de ruta, configuraciones de interfaz de enlace ascendente y descendente, BFD para detección de fallas, ruteo entre VLAN usando SVI, HSRP/VRRP para redundancia de gateway, STP para prevención de loops y endurecimiento de seguridad. Al aprovechar las plantillas de Cisco Catalyst Center, estas configuraciones se pueden automatizar, lo que garantiza implementaciones uniformes y eficientes.

Capa de acceso

Como se ha descrito anteriormente, la capa de acceso conecta los terminales a la red, lo que permite un acceso seguro y fiable. Las configuraciones clave incluyen la configuración de la interfaz de acceso, la configuración de la interfaz de enlace ascendente, VLAN de capa 2, ACL para restringir el acceso al dispositivo y refuerzo de la seguridad.

Consideración del diseño de plantilla

En esta sección se describe cómo diseñar plantillas en Cisco Catalyst Center para generar configuraciones de dispositivos. El Editor de plantillas agiliza el aprovisionamiento permitiendo la creación de plantillas CLI reutilizables y admitiendo la implementación dinámica de configuraciones adaptadas a su red. Catalyst Center admite dos lenguajes de plantilla: Jinja2 y Velocity. Estos lenguajes ayudan en la administración de la configuración de los dispositivos.

Jinja es un popular lenguaje de creación de plantillas fácil de usar y utilizado principalmente con Python para generar contenido dinámico como HTML, XML u otros formatos basados en texto. Permite incrustar variables y estructuras de control (como bucles y condicionales) dentro de las plantillas para crear resultados dinámicos.

Apache Velocity es un motor de creación de plantillas basado en Java que utiliza Velocity Template Language (VTL) para habilitar contenido dinámico en varios documentos, incluidas páginas web, XML o incluso código fuente. Combina datos de objetos Java con plantillas para generar el resultado final.

Este documento cubre solamente las plantillas Jinja2.

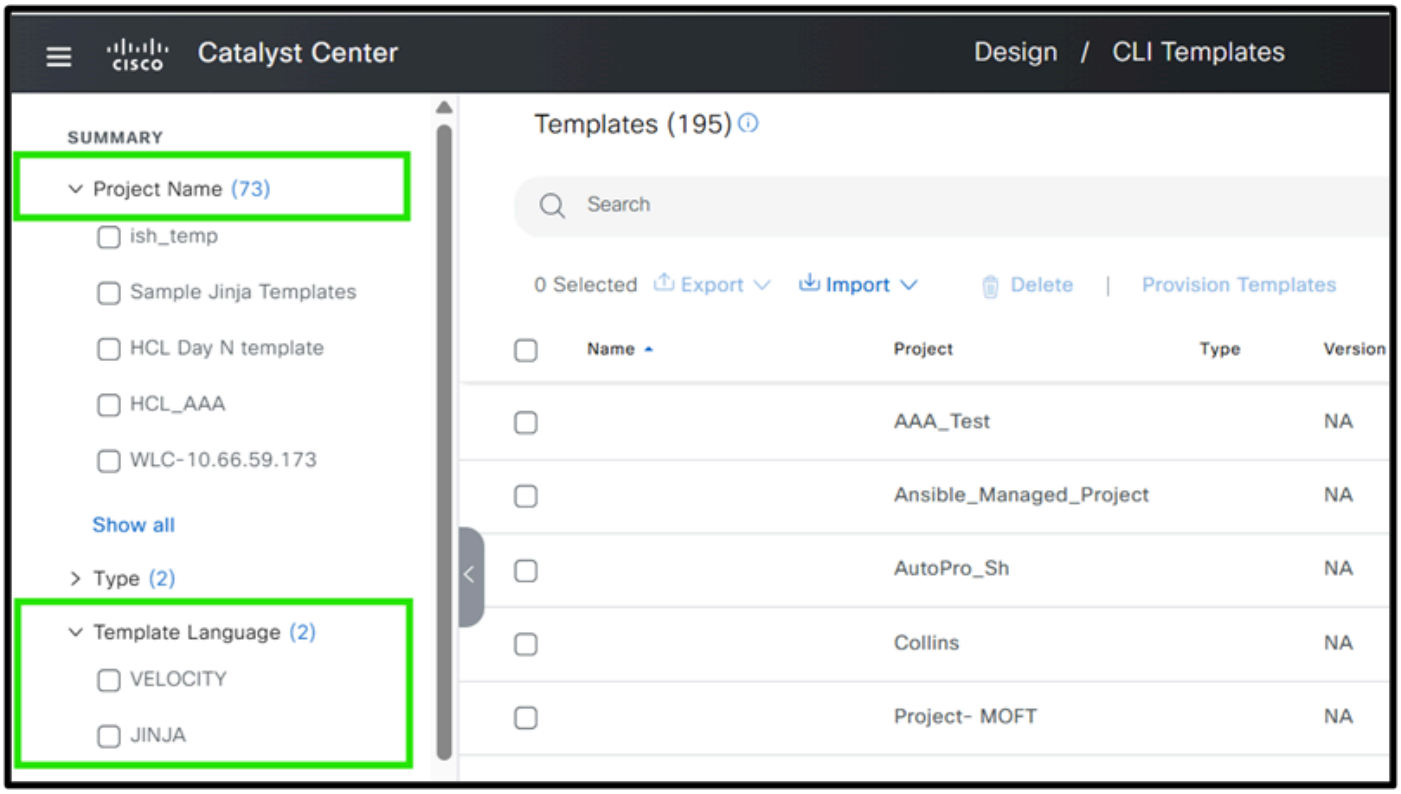


Figura 4: Editor de plantillas de Cisco Catalyst Center

En este documento, utilizamos Jinja2 debido a su flexibilidad. En lugar de una exploración en profundidad de Jinja2, la atención se centra en la aplicación práctica para el diseño de plantillas. Para obtener más información sobre las plantillas Jinja2 en Catalyst Center, consulte el enlace:

https://ciscolearning.github.io/cisco-learning-codelabs/posts/cat-center-j2-part-1/#0

Antes de profundizar en las estrategias de diseño de plantillas para una red de campus de Cisco, es importante utilizar las prácticas recomendadas clave para garantizar la eficacia y la capacidad de gestión al trabajar con plantillas.

Estructura de plantillas y prácticas recomendadas/Directriz para la mejor estrategia

Al automatizar la configuración de los dispositivos de red mediante Cisco Catalyst Center, es fundamental adoptar estrategias estructuradas y prácticas recomendadas. Estos pasos contribuyen a garantizar la uniformidad, la escalabilidad y la facilidad de gestión en toda la infraestructura de red.

Dividir la configuración por función de dispositivo

Comience por categorizar los dispositivos según su función en la topología de red. Las funciones comunes incluyen:

- Núcleo
- Distribución
- Acceso

Ejemplo: Un dispositivo que funcione como switch principal debe tener requisitos de configuración diferentes en comparación con un switch de acceso.

Compartimentación de la configuración en bloques modulares

Dentro de cada función de dispositivo, divida la configuración en bloques modulares agrupando características o configuraciones similares. Este enfoque modular simplifica la automatización, la resolución de problemas y las actualizaciones futuras.

Ejemplos de un dispositivo de núcleo:

- Bloque de configuración OSPF
- Bloque de Configuración BGP
- Bloqueo de políticas de QoS

Identificar bloques de configuración independientes de funciones

Algunos bloques de configuración se aplican universalmente a todos los roles de dispositivos. La identificación y estandarización de estos bloques garantiza prácticas recomendadas y coherencia en toda la red.

Bloques de configuración independientes de funciones comunes:

- Configuración básica: nombre de host, banners de inicio de sesión
- Protocolos de gestión: DHCP, DNS, NTP, SNMP

Políticas de acceso: configuraciones de seguridad estándar

Estos bloques se pueden reutilizar para los dispositivos de núcleo, distribución y acceso, lo que simplifica el proceso de automatización.

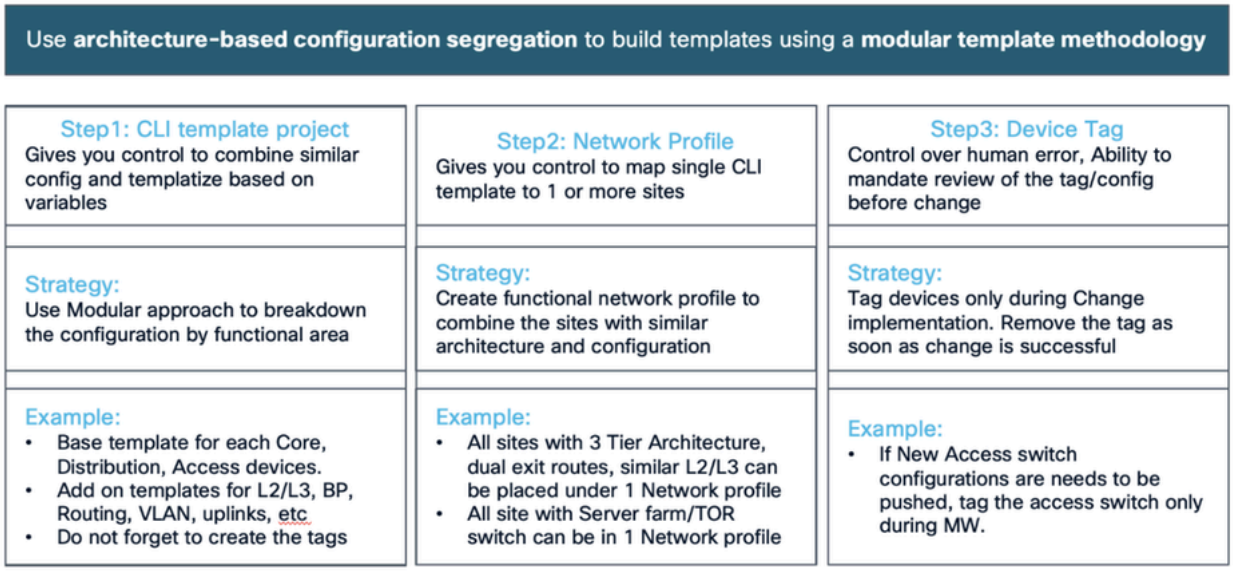


Figura 1: Práctica recomendada con ejemplo

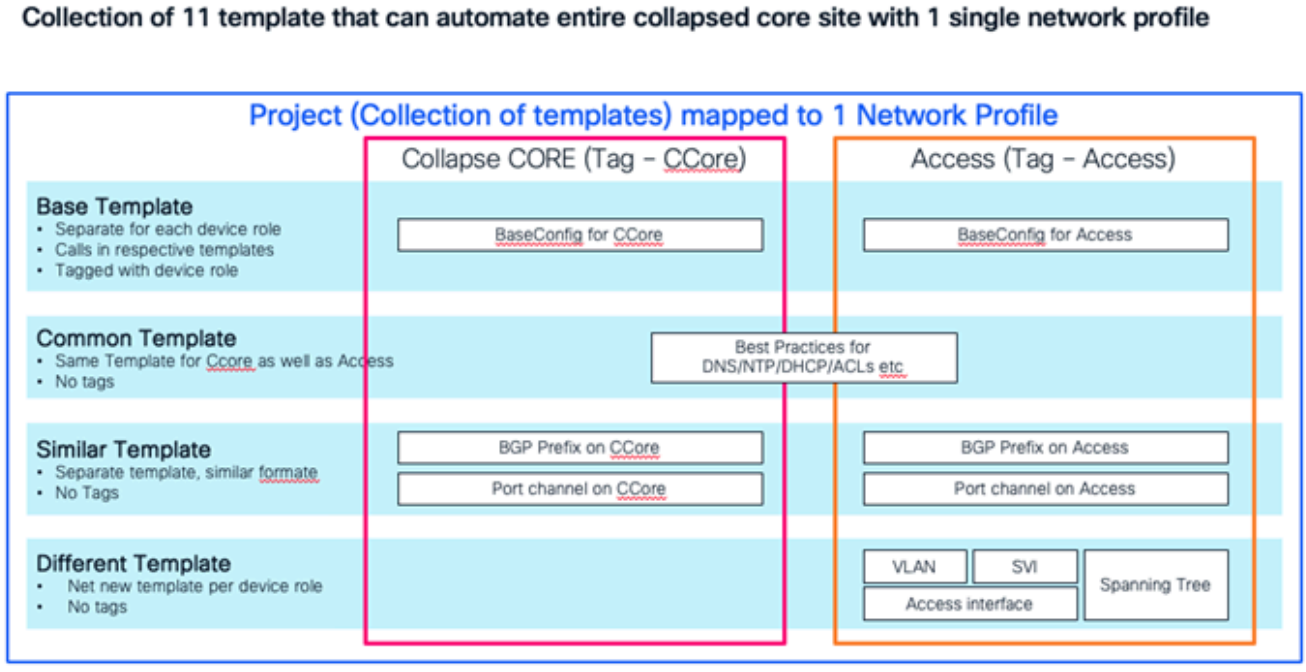


Figura 2: Ejemplo de plantilla de núcleo contraído

Prácticas recomendadas para trabajar con plantillas

Diseño de plantilla modular para la configuración automatizada

Al automatizar las configuraciones de dispositivos en Cisco Catalyst Center, evite incrustar todas las configuraciones en una única plantilla monolítica. En su lugar, adopte un enfoque modular:

Cree una plantilla base que haga referencia a plantillas (módulos) más pequeñas y específicas de un propósito:

Desglose la configuración en módulos lógicos (por ejemplo, configuración de interfaz, protocolos de routing, funciones de seguridad).

Esta estructura hace que las actualizaciones sean más eficientes: los cambios en un módulo específico se reflejan automáticamente dondequiera que se utilice dicho módulo, lo que reduce significativamente los errores y la complejidad.

Ejemplo: Configuración modular para un dispositivo de sucursal

Suponga que está automatizando la configuración de un dispositivo de sucursal.

Plantilla base:

Incluye referencias a plantillas de módulo para áreas de configuración clave.

Pasa las variables según sea necesario a cada módulo para su personalización.

Plantillas de módulo:

interface_settings: Administra las configuraciones de interfaz.

Routing_Protocols: Contiene la configuración de OSPF, EIGRP o BGP.

security_features: Define ACL, reglas de firewall u otras políticas de seguridad.

```
{% include "Branch/Interface Configuration" %}
{% include "Branch/Routing Protocol Configuration" %}
{% include "Branch/Security Configuration" %}

{{ Branch_Interface_Configuration(branch_id) }}
{{ Branch_Routing_Protocol_Configuration(branch_id, ospf_area) }}
{{ Branch_Security_Configuration(branch_id) }}
```

Ejemplo de estructura de plantilla base:

Con esta estructura, cualquier cambio en las configuraciones de routing o seguridad solo tiene que hacerse en sus respectivos módulos, y esos cambios se reflejan instantáneamente dondequiera que se utilice la plantilla base. Esto hace que sus configuraciones sean más manejables y uniformes en todos los routers de sucursales.

Aquí el nombre del proyecto es Sucursal y otros 3 módulos diferentes se definen bajo proyecto. Todos estos elementos se combinan en una plantilla base.

Minimizar variables en la plantilla

Reduzca al mínimo el número de variables de la plantilla para reducir la complejidad y los errores. Menos variables simplifican la implementación, especialmente en redes de gran tamaño, lo que hace que el proceso sea más eficaz y uniforme.

Uso de etiquetas de dispositivo para plantillas

Aproveche las etiquetas de dispositivos de Cisco Catalyst Center, como la ubicación, la función o el sitio, para crear plantillas Jinja2 dinámicas y escalables. Estas etiquetas permiten la lógica condicional, lo que garantiza que se aplican las configuraciones correctas a los dispositivos adecuados. Este enfoque minimiza los errores y simplifica la gestión de plantillas en diversos entornos de red.

Valores Estáticos De Hardcode Donde Sea Posible

Los valores estáticos codificados pueden simplificar las plantillas y mejorar la eficacia de la implementación. Entre los ejemplos más comunes se incluyen las direcciones IP de los servidores DNS, NTP o Syslog, que suelen ser uniformes en todos los dispositivos. Del mismo modo, el uso de ID de VLAN estándar en los switches de acceso permite que estos valores estén codificados de forma rígida, lo que reduce la variabilidad y acelera la implementación.

Adopte un enfoque en dos fases: Plantillas de día 0 y día N

Al incorporar dispositivos mediante servicios como Plug and Play de Cisco, utilice una estrategia de plantillas en dos fases:

Plantillas de día 0: Introduzca las configuraciones básicas para asegurarse de que el dispositivo se puede comunicar con Cisco Catalyst Center.

Plantillas de día N: Implemente funciones y configuraciones avanzadas una vez que el dispositivo esté disponible.

Las prácticas recomendadas hacen posibles plantillas eficientes y escalables que simplifican las implementaciones de redes de campus de Cisco.

Control de espacios en blanco en macros de plantillas Jinja

Al crear plantillas con el lenguaje Jinja, es fundamental controlar cuidadosamente los espacios en blanco y las líneas nuevas, especialmente al representar contenido dinámico en macros. Los espacios en blanco acumulados o las líneas nuevas no deseadas pueden provocar problemas de formato en la salida generada, lo que debe provocar errores de interpretación o de procesamiento descendente. Para resolver esto, Jinja proporciona sintaxis para controlar los espacios en blanco: al colocar un signo menos (-) directamente dentro de los delimitadores ({{- ... -}} o {%- ... -%}), quita los espacios en blanco iniciales o finales alrededor de la expresión. Por ejemplo, reemplazar {{item[1]}} por {{- item[1] -}} garantiza que se quitarán los espacios adicionales o las líneas nuevas cuando se represente la macro. Esta práctica resulta especialmente útil cuando se recorre en iteración listas o se generan archivos de configuración, como se muestra en el fragmento de plantilla. Recomendamos aplicar siempre el control de espacios en blanco en estos escenarios para mantener resultados limpios y predecibles.

Ejemplo (uso recomendado):

```
{% para el elemento de la lista_comodín %}
   {% si elemento[0] == prefijo -%}
      {{- elemento[1] -}}
   {%- endif %}
{%- fin para %}
```

Arquitectura de tres niveles

Este informe técnico comienza con el desarrollo de plantillas para los switches de acceso y los switches de núcleo, y describe los requisitos de cada capa.

Switches de capa de acceso

Los switches de acceso se incorporan mediante Plug and Play y deben requerir una plantilla de día 0. Para obtener más información sobre el proceso Plug and Play en Catalyst Center, consulte el enlace :

https://www.cisco.com/c/en/us/td/docs/cloud-systems-management/network-automation-and-management/catalyst-center/2-3-7/user_guide/b_cisco_catalyst_center_user_guide_237/m_onboard-and-provision-devices-with-plug-and-play.html

Como se ha mencionado anteriormente, Catalyst Center admite los lenguajes de creación de plantillas Velocity y Jinja2. Este documento utiliza Jinja2 para ilustrar la estructura de la plantilla, debido a su flexibilidad. La configuración del switch de capa de acceso se puede implementar mediante la plantilla de día 0 y día N.

Una plantilla básica de día 0 se puede estructurar, consulte el paso 1:

Paso 1: Definir plantilla

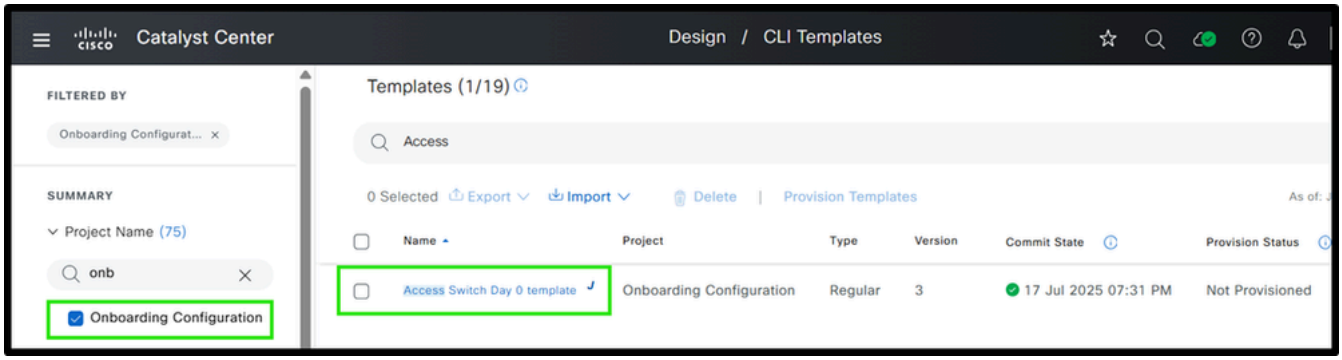
```
username admin privilege 15 password SamplePass123
!
enable secret EnableSecret123
!
ip routing
!
vlan {{ branch_number * 100 + 13 }}
 name SW_MGMT
!
interface vlan {{ branch_number * 100 + 13 }}
 ip address {{ ip_address }} 255.255.255.128
 no ip redirects
 no ip unreachables
 no ip proxy-arp
!
ip route 0.0.0.0 0.0.0.0 {{ nexthop }} name Default-Gateway
!
interface range Te1/1/1 - 2
 switchport
 switchport mode trunk
 no shutdown
!
```

Paso 1: Definir plantilla

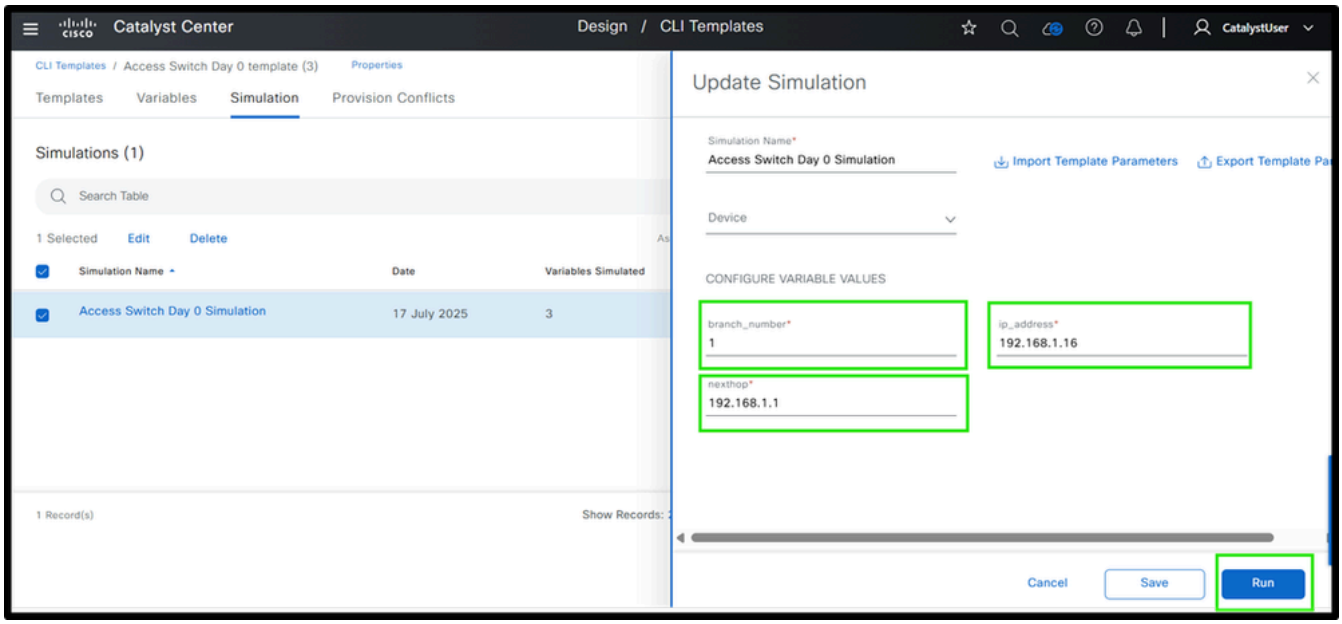
La plantilla simplifica la configuración mediante constantes de codificación rígida como nombre de usuario, contraseña, enable secret y máscara de subred, ya que todos los switches de una sucursal comparten la misma máscara de subred de VLAN de administración. La dirección IP de administración, sin embargo, es única para cada switch y se define como una variable. Se debe proporcionar una estructura de plantilla completa en la plantilla Día N, que utiliza esta plantilla Día 0. En la plantilla de día N, cada función del switch de acceso se gestiona mediante un módulo dedicado; por ejemplo, un módulo gestiona VLAN de capa 2, módulos independientes gestionan interfaces de acceso de enlace ascendente y descendente, otro módulo se centra en la consolidación de la seguridad, etc.

Aunque se prefieren ID de VLAN uniformes, se pueden generar ID variables dinámicamente usando una fórmula basada en el número de sucursal (por ejemplo, Branch 1 = VLAN 113, Branch 2 = VLAN 213). Esto hace que la plantilla sea reutilizable en todas las sucursales. De manera similar, la IP de salto siguiente es una variable, ya que debe diferir por sucursal dependiendo del clúster de distribución conectado.

Paso 2: Realizar simulación y proporcionar variable



Estructura de plantillas de día 0 del switch de acceso con entradas y salidas de simulación



Ej. Entradas de simulación

Siempre se recomienda simular la plantilla antes de la implementación. La captura de pantalla muestra la configuración final después de ingresar las variables.

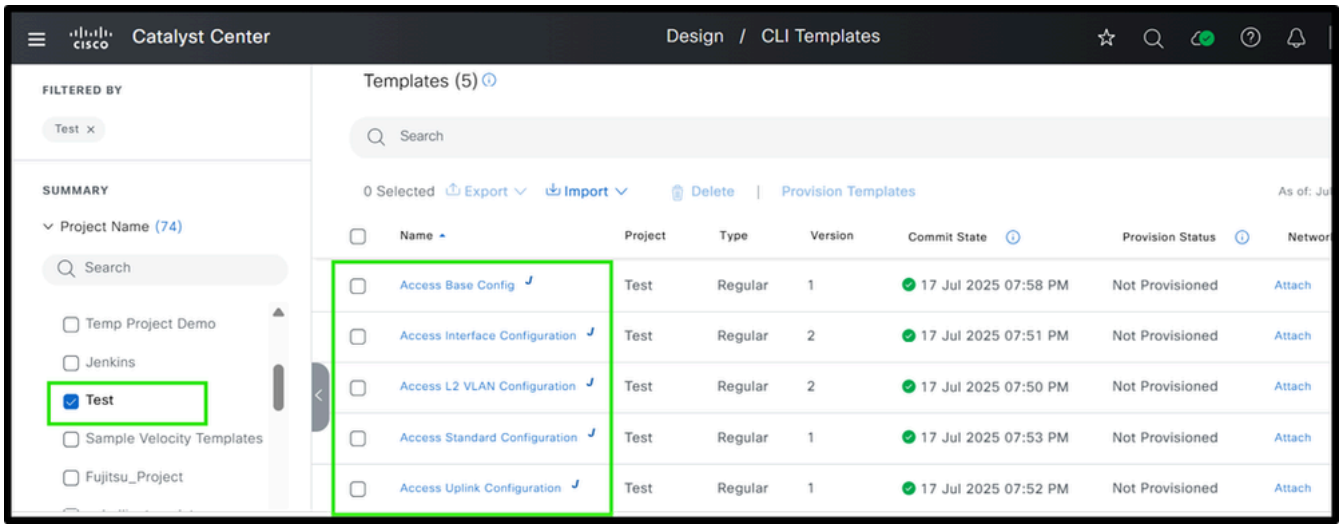
Configuración final después de introducir valores

Ahora, veamos cómo crear una plantilla modular Day N.

La configuración del switch de acceso se puede dividir en varios módulos, todos los cuales se pueden combinar dentro de un módulo base. La plantilla base para los switches de acceso está estructurada como se muestra.

Tanto la plantilla base como sus módulos se crean en un proyecto denominado "Test" de Cisco Catalyst Center.

Paso 1: Definir varias plantillas, incluida la plantilla Base



Estructura de plantillas de día N del switch de acceso

Paso 2: Definir varios módulos

Configuración de base de acceso:

La captura de pantalla muestra un ejemplo de la configuración básica.

```
{% include "Test/Access L2 VLAN Configuration" %}
{% include "Test/Access Interface Configuration" %}
{% include "Test/Access Uplink Configuration" %}
{% include "Test/Access Standard Configuration" %}

{{ Access_L2_VLAN_Configuration(branch_number, is_poe) }}
{{ Access_Uplink_Configuration(branch_number, is_poe)}}
{{ Access_Interface_Configuration(branch_number,  is_poe) }}
{{ Access_Standard_Configuration(branch_number) }}
```

Configuración de base de acceso

Esta plantilla de configuración modular incluye cuatro partes: configuración de VLAN, configuración de interfaz de enlace ascendente, configuración de interfaz de acceso y configuración estándar. Utiliza sólo dos variables: branch_number e is_poe, por lo que resulta sencillo y fácil de gestionar.

Branch_number calcula los ID de VLAN específicos de la sucursal, como se muestra en la plantilla Día 0, y is_poe determina si el switch de acceso es un switch PoE o no PoE. Estas variables se proporcionan durante el aprovisionamiento y se pasan a los módulos para crear las configuraciones correctas, lo que reduce el esfuerzo y mejora la eficacia.

Ahora, revisemos cada módulo para ver cómo contribuyen a generar partes específicas de la

configuración general.

Configuración de VLAN L2 de acceso

```
{% macro Access_L2_VLAN_Configuration (branch_number, is_poe) %}
!
vlan {{ 100 * branch_number + 11 }}
 name DATA_VLAN
!
vlan {{ 100 * branch_number + 12 }}
 name VOICE_VLAN
!
{% if is_poe == 'Yes' %}
vlan {{ 100 * branch_number + 14 }}
 name AP_Mgmt
{% endif %}
!
{% endmacro %}
```

Configuración de VLAN L2 de acceso

Este módulo crea VLAN basadas en el número de sucursal, como se explicó anteriormente. Las VLAN de datos y voz se crean en todos los switches, independientemente de si admiten PoE o no. La VLAN de administración de AP (por ejemplo, 114 para la rama 1) solo se crea si is_poe se establece en "Sí", lo que significa que el switch admite PoE. Si is_poe es "No", la VLAN de administración de AP se omite ya que los switches no PoE no pueden soportar puntos de acceso. Se administra mediante una condición if.

```
{% macro common_access_settings() %}
 switchport port-security maximum 2
 switchport port-security
 switchport port-security violation shutdown
 spanning-tree portfast
 spanning-tree bpduguard enable
 storm-control broadcast level 2.00
 storm-control multicast level 2.00
 storm-control unknown-unicast 2.00
{% endmacro %}

{% macro Access_Interface_Configuration(branch_number, is_poe) %}
!
interface range Gi1/0/1 - 6
{% if is_poe == 'Yes' %}
 description *** AP ***
 switchport mode access
 switchport access vlan {{ 100 * branch_number + 14 }}
{% else %}
 description *** User Ports ***
 switchport mode access
 switchport access vlan {{ 100 * branch_number + 11 }}
 switchport voice vlan {{ 100 * branch_number + 12 }}
{% endif %}
{{ common_access_settings() }}
!
interface range Gi1/0/7 - 24
 description *** User Ports ***
 switchport mode access
 switchport access vlan {{ 100 * branch_number + 11 }}
 switchport voice vlan {{ 100 * branch_number + 12 }}
{{ common_access_settings() }}
!
{% endmacro %}
```

Configuración de interfaz de acceso

Este módulo gestiona la configuración de la interfaz de acceso y utiliza el mismo enfoque que el switch PoE descrito anteriormente. Si la variable is_poe es "Yes", lo que significa que el switch es un switch PoE, los primeros seis puertos (1-6) deben configurarse con la VLAN de administración de AP. De lo contrario, los primeros seis puertos deben configurarse como puertos de acceso de usuario.

Suponiendo que el switch sea un modelo de 24 puertos, los puertos restantes (7-24) siempre se configuran como puertos de acceso de usuario, independientemente de si el switch es PoE o no.

El intervalo de interfaz se ha estandarizado y ya no se considera una variable de entrada, lo que se considera una práctica recomendada para minimizar el número de variables de la plantilla. Además, el módulo incluye una macro denominada common_access_settings, que minimiza el tamaño de la plantilla mediante la consolidación de configuraciones repetidas. Esta macro se llama simplemente dentro de la configuración de la interfaz, evitando la

necesidad de especificarlos varias veces.

Nota: Esta plantilla aplica la misma descripción a todas las interfaces de acceso. Si se necesitan descripciones únicas para cada interfaz, se recomienda enviarlas mediante scripts Python independientes o herramientas de automatización similares.

Revise el módulo que genera las configuraciones para las interfaces de enlace ascendente.

```
{% macro Access_Uplink_Configuration(branch_number, is_poe) %}
{% if is_poe == 'Yes' %}
!
interface range Te 1/1/1 - 2
 switchport
 switchport mode trunk
 switchport trunk allowed vlan {{ branch_number * 100 + 11 }},{{ branch_number * 100 + 12 }},{{ branch_number * 100 + 13 }},{{
branch_number * 100 + 14 }}
 no shutdown
!
{% else %}
!
interface range Te 1/1/1 - 2
 switchport
 switchport mode trunk
 switchport trunk allowed vlan {{ branch_number * 100 + 11 }},{{ branch_number * 100 + 12 }},{{ branch_number * 100 + 13 }}
 no shutdown
!
{% endif %}
{% endmacro %}
```

Acceder a configuración de enlace ascendente

Este módulo genera la configuración para las interfaces de link ascendente y maneja el recorte de VLAN. Si el switch admite PoE, la VLAN de administración de AP se incluye en la lista de VLAN permitidas; de lo contrario, se excluye. Esta lógica se administra mediante la condición if en el código, como se ha descrito anteriormente.

Revise el módulo final, que muestra las configuraciones estándar, incluidas las prácticas recomendadas y la consolidación de la seguridad.

Precaución: Tenga en cuenta que esto es solo con fines ilustrativos y no debe utilizarse como referencia para las configuraciones de red reales, ya que las configuraciones pueden variar según los requisitos específicos

---

```
{% macro Access_Standard_Configuration (branch_number) %}
!
spanning-tree mode rapid-pvst
spanning-tree extend system-id
!
vtp mode off
no errdisable recovery cause all
crypto key generate rsa modulus 2048
!
ip ssh version 2
ip ssh time-out 120
ip ssh source-interface vlan {{ branch_number * 100 + 13 }}
no ip http server
no ip http secure-server
ip http client source-interface vlan {{ branch_number * 100 + 13 }}
!
logging buffered informational
logging host 192.168.1.10
logging host 192.168.2.20
logging source-interface vlan {{ branch_number * 100 + 13 }}
!
ntp authentication
ntp authentication-key 10 md5 Netw0rkAuthKey
ntp source vlan {{ branch_number * 100 + 13 }}
ntp server 192.168.3.1 key 10
ntp server 192.168.3.2 key 10
!
snmp-server enable traps
snmp-server trap-source vlan {{ branch_number * 100 + 13 }}
snmp-server group NMSNWDEVICE v3 priv access SNMPHOST
snmp-server user netadmin NMSNWDEVICE v3 auth sha AuthKey123 priv aes 128 PrivKey123
!
ip access-list standard SNMPHOST
permit 192.168.4.0 0.0.0.255
!
ip access-list standard VTYACL
permit 192.168.5.10
```

Parte 1: Configuración estándar de acceso

```
permit 192.168.5.11
!
aaa new-model
ip tacacs source-interface vlan {{ branch_number * 100 + 13 }}
tacacs server TACACS_1
 address ipv4 192.168.6.1
 key TACACSKey123
 timeout 4
tacacs server TACACS_2
 address ipv4 192.168.6.2
 key TACACSKey123
 timeout 4
aaa group server tacacs+ TACACS-SERVER
 server name TACACS_1
 server name TACACS_2
!
aaa authentication login default group TACACS-SERVER local
aaa authorization exec default group TACACS-SERVER local
aaa accounting exec default start-stop group TACACS-SERVER
!
line console 0
login authentication default
exec-timeout 5 0
!
line vty 0 15
login authentication default
access-class VTYACL in
exec-timeout 5 0
!
banner login ^
********************************** WARNING ************************************
All systems/network should be used/accessed by authorized persons only
  If you are not authorized to do so, you should log off immediately
    Access to and usage of this system /network may be monitored
    All users must comply with information security policies
        Any Violation may lead to disciplinary action.
*******************************************************************************^
{% endmacro %}
```

Parte 2: Configuración estándar de acceso

Este módulo genera una configuración estándar que incorpora prácticas recomendadas, refuerzo de la seguridad y funciones clave para la gestión segura de dispositivos. La mayoría de los valores están codificados para mantener la coherencia entre las sucursales, excepto branch_number, que se utiliza para calcular la VLAN de administración para los switches en cada sucursal y sirve como interfaz de origen para varias configuraciones.

Paso 3: Realice una simulación antes de configurar los switches. Sólo se debe simular la configuración básica.

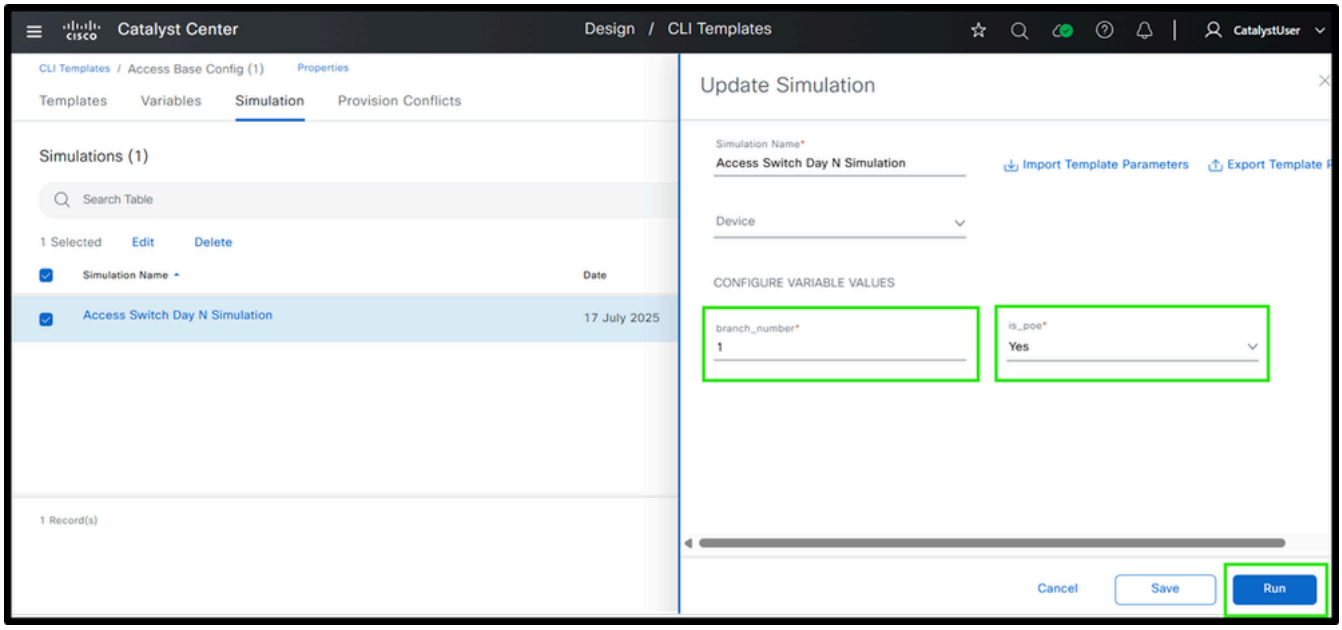
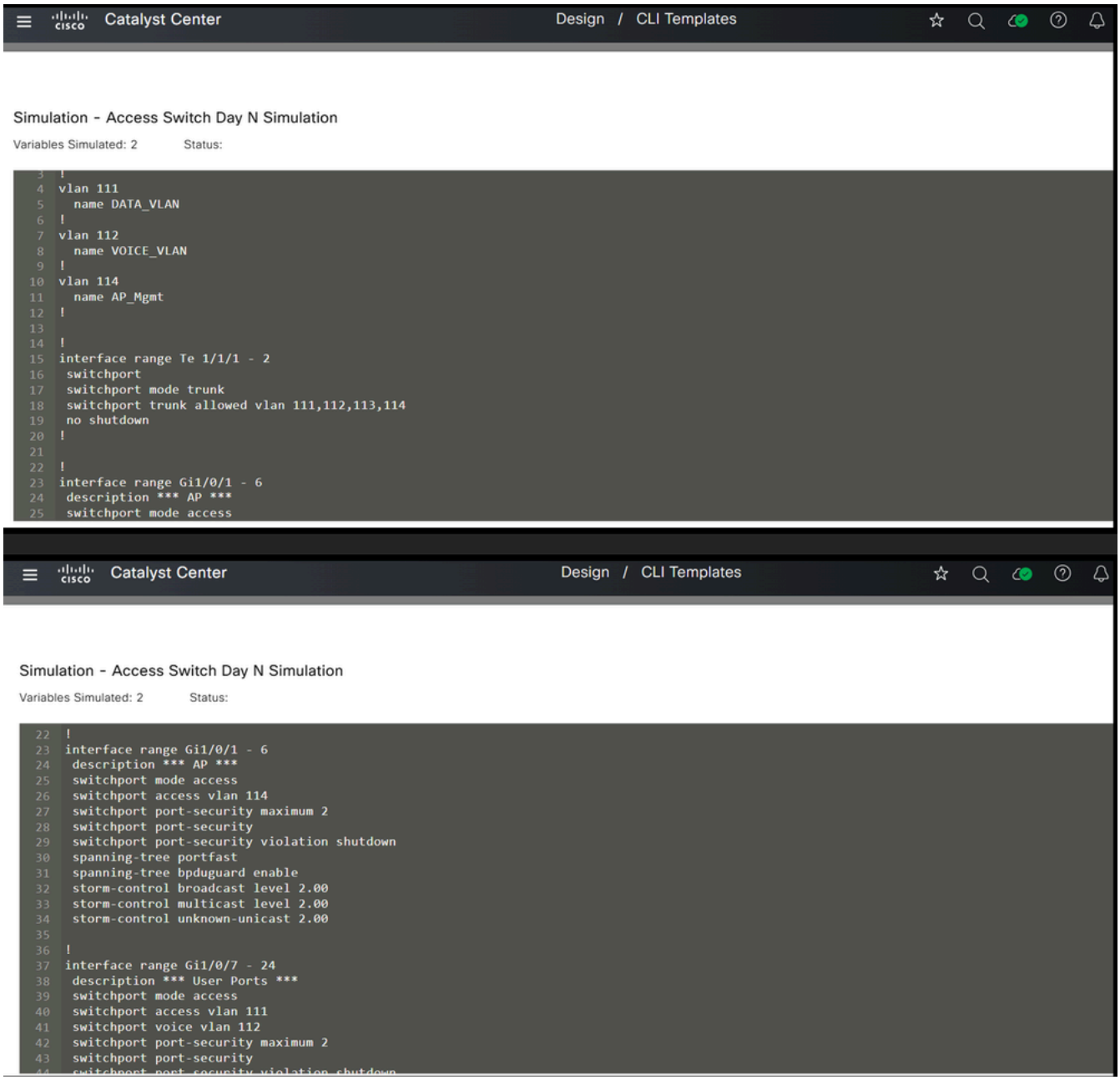


Figura 7: Entradas y salidas de la simulación de la plantilla del día N del switch de acceso



Simulación

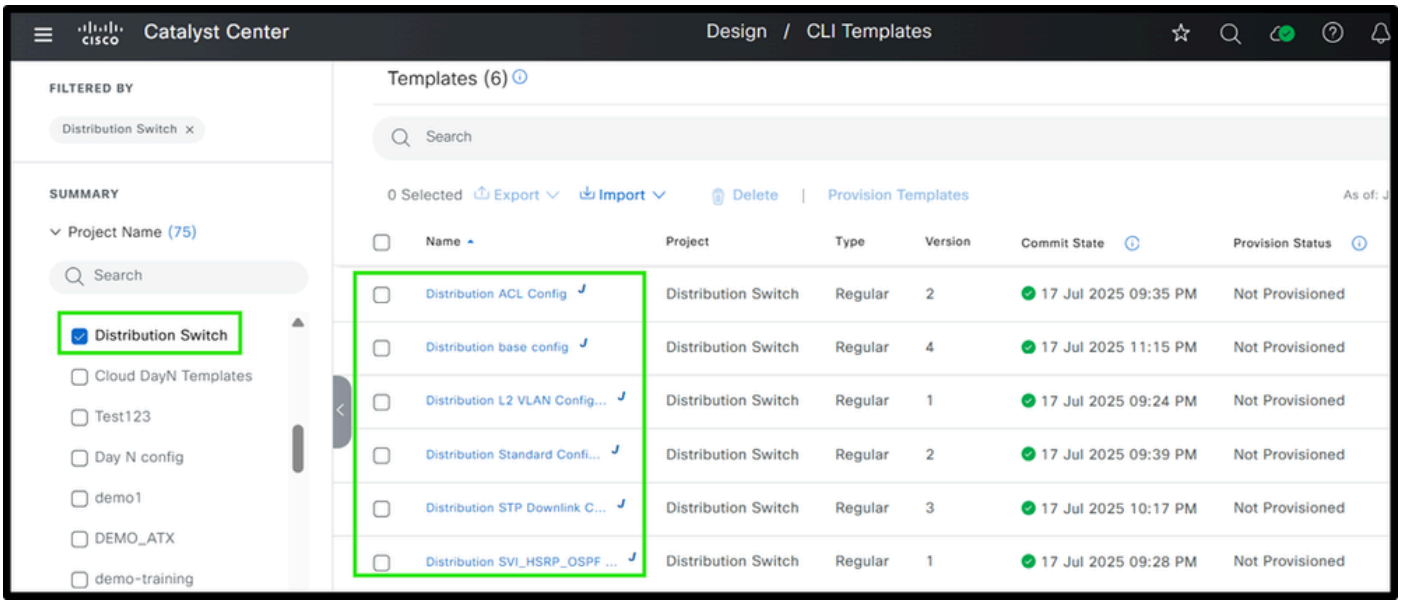
Así es como se pueden utilizar las plantillas en la capa de acceso para generar configuraciones.

Ahora echemos un vistazo a los dispositivos de la capa de distribución para ver cómo se les pueden aplicar las plantillas.

Switches de capa de distribución

Ahora diseñamos una plantilla modular para los switches de distribución. La plantilla base y sus módulos forman parte del proyecto "Switch de distribución" en Cisco Catalyst Center.

Paso 1: Estructura de plantilla de switch de distribución



Ej. Plantillas de distribución

Paso 2: Definir cada módulo

La configuración básica proporcionada define cada módulo y se hace referencia a todos ellos.

```
{% include "Distribution Switch/Distribution L2 VLAN Configuration" %}
{% include "Distribution Switch/Distribution STP Downlink Config" %}
{% include "Distribution Switch/Distribution SVI_HSRP_OSPF Config" %}
{% include "Distribution Switch/Distribution ACL Config" %}
{% include "Distribution Switch/Distribution Standard Configuration" %}

{{ Distribution_L2_VLAN_Configuration(branch_number, is_poe) }}}}
{{ Distribution_STP_Downlink_Config(branch_number, is_poe, distribution_number) }}
{{ Distribution_SVI_HSRP_OSPF_Config(branch_number, is_poe, distribution_number) }}
{{ Distribution_ACL_Config(branch_number) }}
{{ Distribution_Standard_Config() }}
```

Ej. Módulos de plantilla de base de distribución

De forma similar a los switches de acceso, todas las plantillas se crean dentro del proyecto 'Switch de distribución' y se hace referencia a ellas en la plantilla base. Aunque algunas plantillas son idénticas a las utilizadas para los switches de acceso, esta sección explica las diferencias específicas de los switches de distribución. El módulo "Configuración de VLAN de capa 2 de distribución" es idéntico al descrito anteriormente para los switches de acceso. Verifique el módulo Access L2 VLAN Configuration que proporciona esta información. Genera las VLAN necesarias en función de los valores de entrada proporcionados para las variables.

Ahora revise el módulo "Distribution STP Downlink Config", que se encarga de la generación de las configuraciones de spanning tree y uplink para los switches de distribución.

```
{% macro Distribution_STP_Downlink_Config (branch_number, is_poe, distribution_number) %}
!
spanning-tree mode rapid-pvst

{% set base_vlan = branch_number * 100 %}
{% set vlans = [base_vlan + 11, base_vlan + 12, base_vlan + 13] %}

{% if is_poe == 'Yes' %}
  {% set vlans = vlans + [base_vlan + 14] %}
{% endif %}

{% if distribution_number == 1 %}
   {% set stp_priority = 4096 %}
{% else %}
   {% set stp_priority = 8192 %}
{% endif %}

spanning-tree vlan {{ vlans | join(',') }} priority {{ stp_priority }}
!
interface range TWE 1/0/1 - 2
 switchport
 switchport mode trunk
 switchport trunk allowed vlan {{ vlans | join(',') }}
 no shutdown
!
{% endmacro %}
```

Configuración de enlace descendente STP de distribución

Aquí se está utilizando la funcionalidad de macro Jinja2 , a la que se hace referencia en el módulo basado. Esta estructura ayuda a construir un enfoque modular.

Este módulo configura las interfaces de protocolo de árbol de extensión (STP) y de enlace descendente según el "branch_number" y si el switch está habilitado para PoE. La variable "branch_number" se utiliza para generar VLAN base únicas para cada sucursal, asegurando VLAN distintas, similar al enfoque ya resaltado para los switches de acceso. Si el switch está habilitado para PoE ("is_poe" == 'Yes'), se agrega a la lista una VLAN adicional, como la VLAN de administración de AP. La variable "distribution_number" determina la prioridad STP, configurando 4096 para la Distribución 1 (convirtiéndolo en el root bridge preferido) y 8192 para los switches de distribución secundarios. Por último, las VLAN adecuadas se aplican a la interfaz troncal, lo que garantiza que solo se permitan las VLAN relevantes en función de si el switch está habilitado para PoE.

Ahora revise el módulo "Distribution SVI_HSRP_OSPF Config", que se centra en la configuración de SVIs, HSRP y OSPF para un ruteo de red y redundancia eficientes.

```
{% macro Distribution_SVI_HSRP_OSPF_Config (branch_number, is_poe, distribution_number) %}
!
interface loopback0
ip address {{ loopback_ip }} 255.255.255.255
!
router ospf 1
router-id {{ loopback_ip }}
!
key chain HSRP_KEY
 key 0
  key-string cisco@7875
!
interface vlan {{ 100 * branch_number + 11 }}
description Data_Endpoints
ip address 172.17.{{ (branch_number - 1) * 16 }}.{{ distribution_number + 1 }} 255.255.240.0
standby bfd
standby version 2
standby {{ 100 * branch_number + 11 }} ip 172.17.{{ (branch_number - 1) * 16 }}.1
{% if distribution_number == 1 %}
standby {{ 100 * branch_number + 11 }} priority 255
{% else %}
standby {{ 100 * branch_number + 11 }} priority 250
{% endif %}
standby {{ 100 * branch_number + 11 }} authentication md5 key-chain HSRP_KEY
standby {{ 100 * branch_number + 11 }} preempt delay minimum 120
no ip redirects
no ip unreachables
no ip proxy-arp
ip ospf 1 area 0
bfd interval 100 min_rx 100 multiplier 3
!
! uplink interfaces
interface TWE1/1/1
 no switchport
 ip address {{ twe1_1_1_ip }} 255.255.255.0
 ip ospf 1 area 0
 no shutdown
!
interface TWE1/1/2
 no switchport
 ip address {{ twe1_1_2_ip }} 255.255.255.0
 ip ospf 1 area 0
 no shutdown
!
{% endmacro %}
```

Config. SVI_HSRP_OSPF de distribución

Este módulo, Distribution_SVI_HSRP_OSPF_Config, ayuda a configurar las interfaces SVI, HSRP, OSPF y de enlace ascendente para los switches de distribución. En este ejemplo, nos centramos en la SVI para subredes de datos, pero el mismo método se puede utilizar para otras SVI, como la voz o la gestión.

Si la planificación de la dirección IP para las subredes de datos ya se ha realizado, las direcciones IP se pueden calcular automáticamente para cada SVI en función de las variables branch_number y distribution_number. Por ejemplo, si la sucursal 1 tiene la subred 172.17.0.0/20, la sucursal 2 tiene 172.17.16.0/20, y la sucursal 3 tiene 172.17.32.0/20, la dirección IP de la puerta de enlace es 172.17.x.1 (donde x es el número de sucursal). La segunda IP para el primer switch de distribución es 172.17.x.2, y la tercera IP para el segundo switch de distribución es 172.17.x.3. De esta manera, las direcciones IP se calculan automáticamente, reduciendo los errores y simplificando el proceso.

A la interfaz de loopback se le asigna una IP de la variable loopback_ip, que sirve como ID del router OSPF para garantizar un ruteo estable y consistente a través de la red. En la configuración OSPF, esta IP de loopback se utiliza como ID del router y las interfaces relevantes se agregan al área OSPF 0. Para HSRP, se establecen los valores de prioridad: 255 para el primer switch de distribución y 250 para el segundo, lo que garantiza una conmutación por fallo adecuada. Además, la autenticación HSRP se configura mediante una cadena de claves (HSRP_KEY) para mejorar la seguridad.

Para mantener la configuración limpia y manejable, algunos valores están codificados. Por ejemplo, la máscara de subred (255.255.240.0) y ciertas configuraciones HSRP (como la versión y BFD) son las mismas en todas las sucursales, lo que reduce el número de variables. Esto hace que la configuración sea más sencilla, más fácil de aplicar y menos probable que tenga errores. Por último, las interfaces de link ascendente se configuran con IP y se agregan al área OSPF 0 para un ruteo adecuado entre los switches. Este enfoque hace que el proceso de configuración sea más fácil de gestionar y menos propenso a errores, a la vez que es flexible para las diferentes sucursales.

Ahora revise el módulo "Configuración de ACL de Distribución", que proporciona segmentación en la capa de distribución.

```
{% macro Distribution_ACL_Config (branch_number) %}
!
ip access-list extended BLOCK_BRANCH
 deny ip 172.17.{{ 16 * (branch_number - 1) }}.0 0.0.15.255 172.16.{{ 16 * (branch_number - 1) }}.0 0.0.15.255
 deny ip any host 239.255.255.250
 permit ip any any
!
interface vlan {{ 100 * branch_number + 11 }}
 ip access-group BLOCK_BRANCH in
!
{% endmacro %}
```

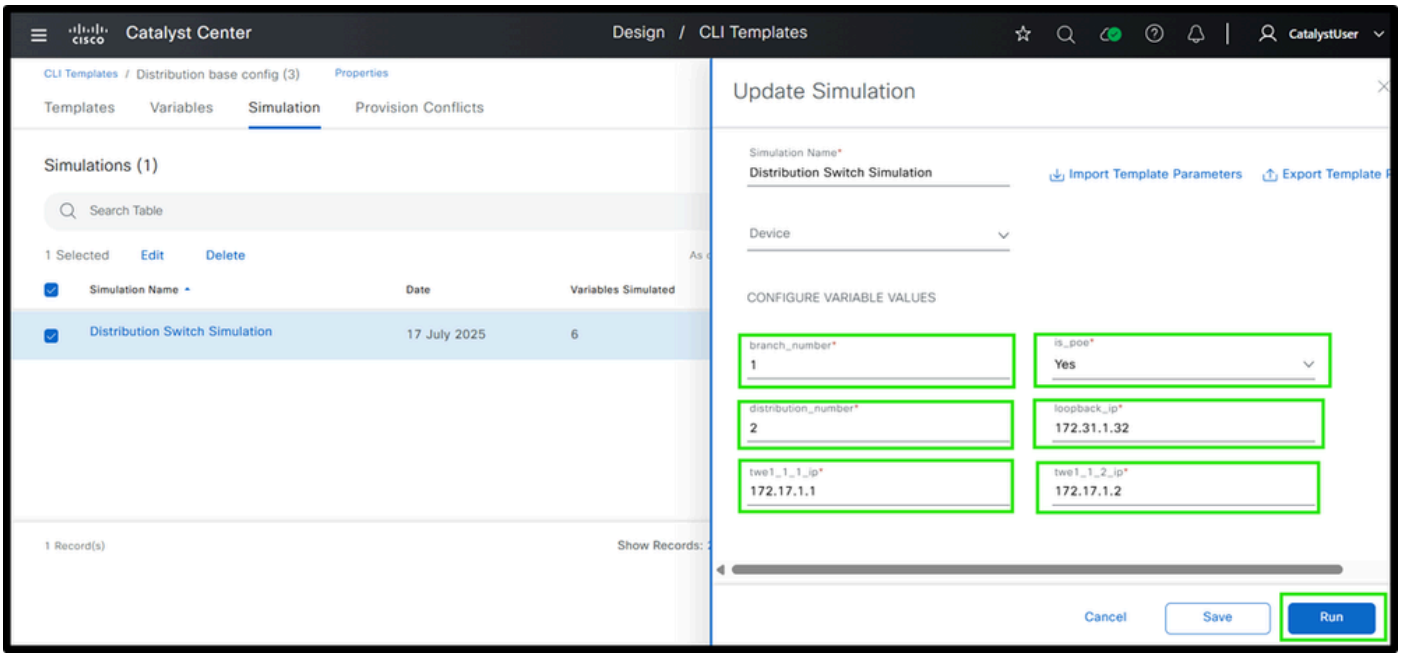
Configuración de ACL de distribución

Este módulo demuestra la segmentación en la capa de distribución usando una plantilla Jinja2. Utiliza la variable branch_number para calcular dinámicamente las direcciones de subred, lo que

permite configuraciones de ACL automatizadas y escalables. Para cada sucursal, la ACL bloquea la comunicación entre la subred 1 (172.17.X.0) y la subred 2 (172.16.X.0) denegando el tráfico IP entre estos rangos. También niega el tráfico a la dirección multicast 239.255.255.250, mientras permite el resto del tráfico. La interfaz VLAN se asigna dinámicamente en función del número de sucursal, y la ACL se aplica de forma entrante en esa interfaz. Este enfoque automatizado garantiza una segmentación eficaz por sucursal, reduce los errores de configuración manual y simplifica la aplicación de políticas de red.

Por último, el último módulo, "Configuración estándar de distribución", es casi idéntico al descrito en el módulo Configuración estándar de acceso (consulte esa sección para obtener más información). Incluye prácticas recomendadas, refuerzo de la seguridad y funciones clave para la gestión segura de dispositivos. La única diferencia reside en la interfaz de origen: en la plantilla del switch de acceso, se define como VLAN {{ branch_number * 100 + 13 }}, mientras que en la configuración del switch de distribución, se puede codificar como Loopback0.

Paso 3: Realice la simulación antes de implementar la configuración.



(1) Entradas y salidas de simulación de plantilla de switch de distribución

(2) Entradas y salidas de simulación de plantilla de switch de distribución

(3) Entradas y salidas de simulación de plantilla de switch de distribución

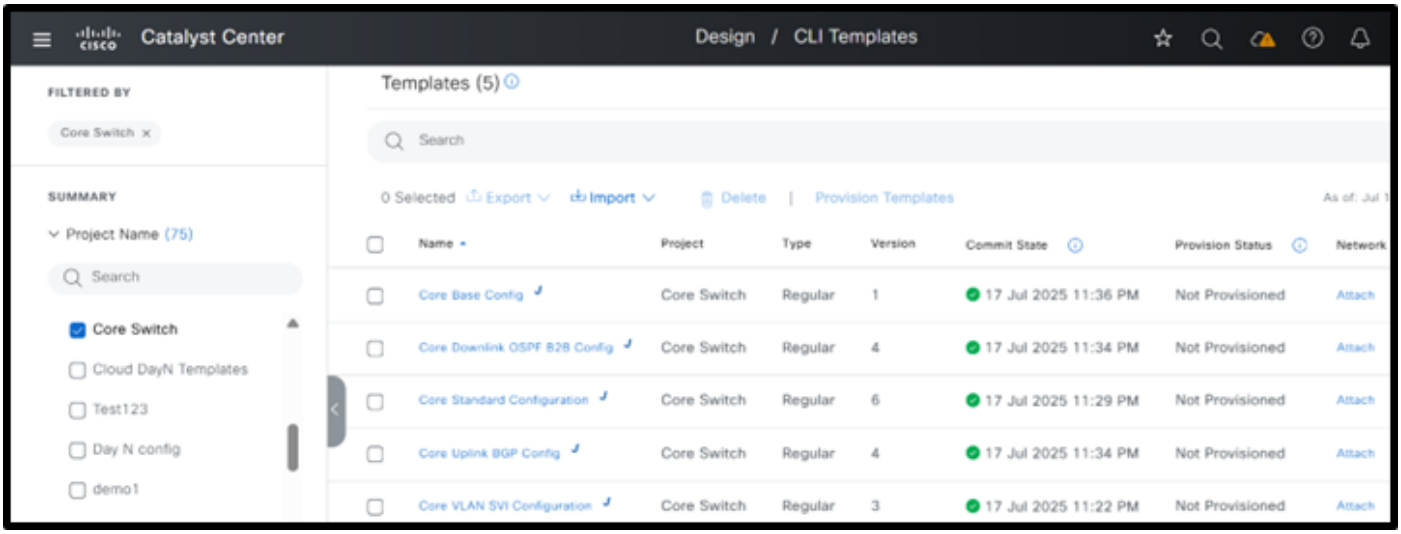
(4) Entradas y salidas de simulación de plantilla de switch de distribución

Así es como se pueden utilizar las plantillas en la capa de distribución para generar configuraciones. Ahora echemos un vistazo a los dispositivos de la capa principal para ver cómo se pueden aplicar las plantillas.

Switches de capa de núcleo

Ahora diseñe una plantilla modular para los switches de núcleo. La plantilla básica y sus módulos forman parte del proyecto "Switch principal" en Cisco Catalyst Center. Consulte la plantilla base en el paso 1.

Paso 1: Definir varias estructuras de switches de núcleo



Estructura de plantilla de switch principal

Paso 2: Definir varios módulos

```
{% include "Core Switch/Core VLAN SVI Configuration" %}
{% include "Core Switch/Core Downlink OSPF B2B Config" %}
{% include "Core Switch/Core Uplink BGP Config" %}
{% include "Core Switch/Core Standard Configuration" %}

{{ Core_VLAN_SVI_Configuration () }}
{{ Core_Downlink_OSPF_B2B_Config () }}
{{ Core_Uplink_BGP_Config () }}
{{ Core_Standard_Config () }}
```

Configuración básica del núcleo

La mayoría de las configuraciones de los switches de núcleo son similares en todas las sucursales, por lo que los valores comunes se pueden codificar. Por lo general, sólo cambian las direcciones IP, que se pueden establecer mediante variables. Dado que cada sucursal normalmente solo tiene dos switches de núcleo, la gestión de estas variables es sencilla. Incluso si algunas sucursales tienen más switches de núcleo, su número sigue siendo inferior al número de switches de acceso o distribución. Por este motivo, como práctica recomendada, es más importante minimizar las variables de los switches de acceso y distribución, ya que se utilizan en grandes cantidades y tener demasiadas variables puede hacer que la configuración lleve más tiempo.

Ahora comience con el primer módulo: "Configuración de SVI de VLAN principal". En este ejemplo, los switches de núcleo se colocan detrás de un firewall y deben establecer un peering eBGP con él. Este módulo es responsable de generar las VLAN y las SVI correspondientes necesarias para el peering eBGP y la vecindad OSPF. Se supone que el firewall funciona en una configuración activa/en espera.

```
{% macro Core_VLAN_SVI_Configuration () %}
!
vlan 2001
 name eBGP_peering_to_FW
!
vlan 2002
 name OSPF_neighborship
!
interface vlan 2001
 description eBGP Peering to Firewall
 ip address {{ VLAN2001_IP }} 255.255.255.248
 bfd interval 100 min_rx 100 multiplier 3
 no ip redirects
 no ip unreachables
 no ip proxy-arp
!
interface vlan 2002
 description OSPF neighborship to Core SW 2
 ip address {{ VLAN2002_IP }} 255.255.255.248
 bfd interval 100 min_rx 100 multiplier 3
 ip ospf 1 a 0
 no ip redirects
 no ip unreachables
 no ip proxy-arp
!
{% endmacro %}
```

Configuración de SVI de VLAN principal

Este módulo, como se explicó anteriormente, crea las VLAN necesarias y las SVI asociadas para establecer relaciones de vecinos OSPF y BGP. Todos los parámetros, excepto las direcciones IP de SVI, están codificados, incluida la máscara de subred si se alinea con el plan de direcciones IP. Este método ayuda a limitar las variables y reduce la posibilidad de errores de configuración.

Ahora, revisemos el módulo "Core Downlink OSPF B2B Config", que genera configuraciones para interfaces de enlace descendente, OSPF y enlaces adosados entre el switch principal 1 y el switch principal 2.

```
{% macro Core_Downlink_OSPF_B2B_Config () %}
!
interface loopback0
ip address {{ loopback_ip }} 255.255.255.255
!
router ospf 1
router-id {{ loopback_ip }}
default-information originate
!
! downlink interfaces
interface TWE1/0/1
 ip address {{ twe1_0_1_ip }} 255.255.255.0
 ip ospf 1 area 0
 no shutdown
!
interface TWE1/0/2
 ip address {{ twe1_0_2_ip }} 255.255.255.0
 ip ospf 1 area 0
 no shutdown
!
interface TWE1/0/24
 description Towards_Core_SW
 switchport mode trunk
 switchport trunk allowed vlan 2001,2002
 channel-group 10 mode active
 spanning-tree portfast trunk
 no shutdown
!
interface TWE1/0/48
 description Towards_Core_SW
 switchport mode trunk
 switchport trunk allowed vlan 2001,2002
 channel-group 10 mode active
 spanning-tree portfast trunk
 no shutdown
!
interface Port-channel10
 description Towards_Core_SW
 switchport mode trunk
 switchport trunk allowed vlan 2001,2002
 spanning-tree portfast trunk
 no shutdown
!
{% endmacro %}
```

Configuración B2B OSPF de enlace descendente principal

De forma similar al módulo anterior, la mayoría de los valores de este módulo están codificados para minimizar el número de variables. Solamente las direcciones IP para las interfaces de loopback y de link descendente son variables. Además, los canales de puerto adosados y las VLAN están estandarizados en las diferentes sucursales.

Ahora, revisemos el módulo "Core Uplink BGP Config", que genera configuraciones BGP y administra enlaces ascendentes conectados a los firewalls.

```
{% macro Core_Uplink_BGP_Config () %}
!
router bgp {{ AS_Number }}
bgp log-neighbor-changes
bgp router-id interface Loopback0
bgp graceful-restart
!
! eBGP Peering with Firewall
neighbor {{ BGP_NEIGHBOR }} remote-as {{ REMOTE_AS }}
neighbor {{ BGP_NEIGHBOR }} description eBGP Peering with Firewall
neighbor {{ BGP_NEIGHBOR }} update-source vlan 2001
neighbor {{ BGP_NEIGHBOR }} fall-over bfd
aggregate-address {{ AGGREGATE_IP }} {{ AGGREGATE_MASK }} summary-only
!
address-family ipv4
network {{ loopback_ip }} mask 255.255.255.255
neighbor {{ BGP_NEIGHBOR }} activate
exit-address-family
!
! Redistribute OSPF into BGP
redistribute ospf
!
! Uplink intefaces
interface TWE1/0/23
 description Towards_Firewall
 switchport mode access
 switchport access vlan 2001
 channel-group 10 mode active
 spanning-tree portfast
 no shutdown
!
interface TWE1/0/47
 description Towards_Firewall
 switchport mode access
 switchport access vlan 2001
 channel-group 10 mode active
 spanning-tree portfast
 no shutdown
!
interface Port-channel10
 description Towards_Firewall
 switchport mode access
 switchport access vlan 2001
 spanning-tree portfast
 no shutdown
!
{% endmacro %}
```

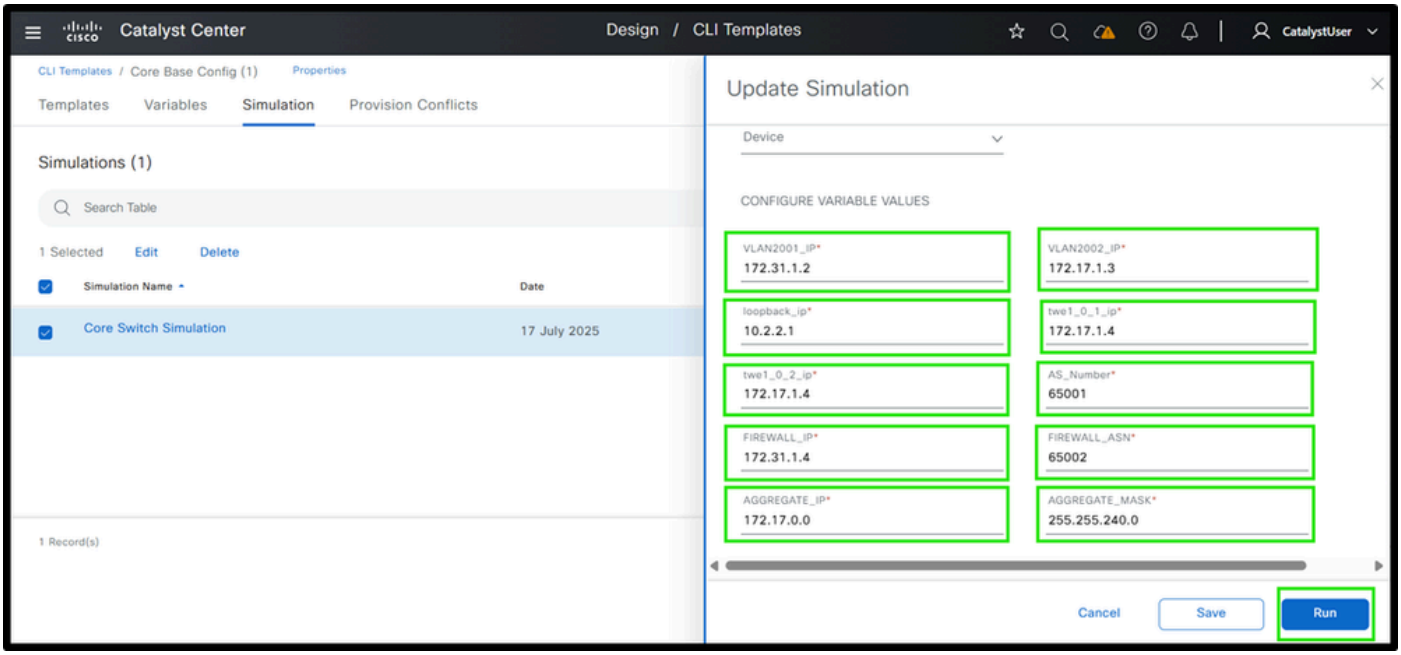
Configuración de BGP de enlace ascendente principal

Este módulo genera la configuración de BGP necesaria para establecer una relación de vecino eBGP con el firewall. Como se muestra anteriormente, la mayoría de los valores están

codificados, ya que siguen siendo coherentes entre las distintas ramas. Sólo las direcciones IP y los números AS, que pueden variar para cada sucursal, se toman como variables de entrada y se utilizan para generar la configuración necesaria. La mayoría de las demás configuraciones se han estandarizado para minimizar el número de variables. Las interfaces de enlace ascendente conectadas al firewall se especifican junto con la VLAN utilizada para la vecindad eBGP, generada por el módulo anterior.

Por último, el último módulo, "Core Standard Configuration", es casi idéntico al que se describe en Access Standard Configuration (consulte esa sección para obtener más detalles). Incluye prácticas recomendadas, refuerzo de la seguridad y funciones clave para la gestión segura de dispositivos. De acuerdo con la configuración del switch de distribución, la interfaz de origen también se puede establecer en loopback0 en este módulo, y este valor se puede codificar.

Paso 3: Realizar simulación

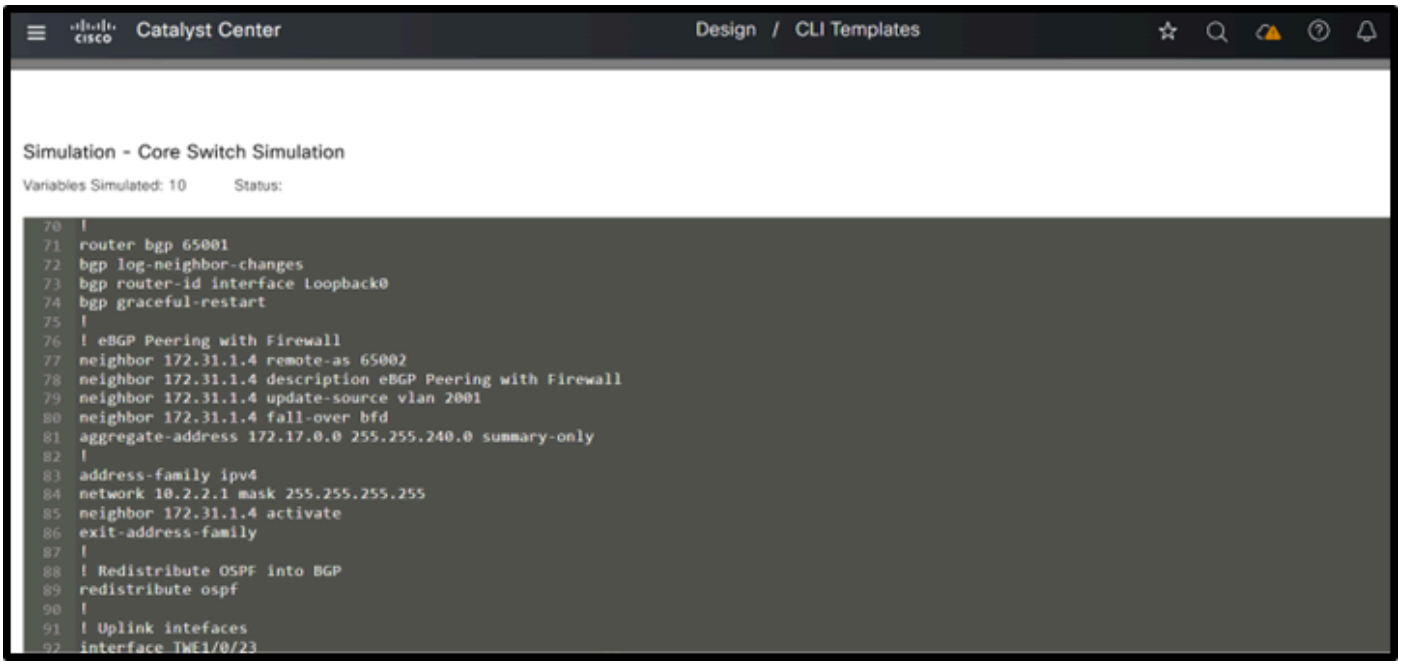


(1) Entradas y salidas de simulación de plantilla de switch principal

(2) Entradas y salidas de simulación de plantilla de switch principal

(3) Entradas y salidas de simulación de plantilla de switch principal

(4) Entradas y salidas de simulación de plantilla de switch principal



(5) Entradas y salidas de simulación de plantilla de switch principal

Aquí se completa la explicación detallada del diseño de plantillas para la arquitectura de tres niveles, con una descripción tanto de la estructura como de la configuración de cada módulo.

Todos estos módulos utilizan las prácticas recomendadas que se han explicado anteriormente.

Nota: Al diseñar plantillas para una arquitectura de núcleo colapsada, consulte las explicaciones proporcionadas para la arquitectura de tres niveles. La estructura de las

---

plantillas sigue siendo la misma; sin embargo, las funciones que antes se implementaban por separado en las capas de núcleo y distribución ahora se combinan en la capa de núcleo contraída. El mismo enfoque de plantilla modular se puede utilizar aquí también, creando una plantilla base y haciendo referencia a los módulos relevantes dentro de ella.

Summary

La arquitectura tradicional de campus de 3 niveles a menudo se basa en una amplia configuración manual en las capas de núcleo, distribución y acceso. Este enfoque no solo lleva mucho tiempo, sino que también es propenso a errores humanos. La ausencia de automatización y de gestión centralizada aumenta considerablemente los costes operativos, lo que dificulta la escalabilidad y la gestión eficaz de las redes de campus dinámicas y modernas. A través de la plantilla de CLI de Catalyst Center, las configuraciones de funciones se pueden automatizar para las redes LAN tradicionales. Es importante utilizar el enfoque modular al aprovisionar los dispositivos. Los módulos se pueden basar en las distintas funciones utilizadas en las distintas capas. Y finalmente enlazar todos estos módulos al módulo base.

Llamada a la acción

Invitamos a las organizaciones a adoptar la metodología de plantillas modulares presentada en este informe técnico como una práctica recomendada para estandarizar las configuraciones de switches y optimizar las operaciones de red en arquitecturas de núcleo de tres niveles y contraídas.

- Al implementar plantillas modulares, los equipos de red pueden:
- Mejore la eficacia operativa mediante prácticas de configuración uniformes y reproducibles.
- Minimice los errores humanos y reduzca el tiempo de resolución de problemas.
- Lograr una mayor escalabilidad para adaptarse al crecimiento y a las necesidades empresariales en constante evolución.
- Garantizar la coherencia de la configuración en diversos entornos.

Este enfoque no solo simplifica la gestión diaria, sino que también permite implementaciones más rápidas, simplifica los ciclos de actualización y mejora la alineación con los requisitos de seguridad y cumplimiento. La adopción de plantillas modulares posiciona su red para disfrutar de agilidad, resistencia y éxito a largo plazo en un panorama de TI en constante cambio.

Para ver demostraciones prácticas, aprenda más sobre plantillas , por favor vea la serie de YouTube

1 Cómo crear y administrar plantillas en Catalyst Center

https://youtu.be/SyUqEEcwye0

2 Cómo utilizar variables de enlace del sistema en plantillas CLI en Catalyst Center

https://youtu.be/gV1QBuHYJdo

Autores

Naveen Kumar, Arquitecto de prestación de servicios al cliente, Cisco Customer Experience

Risabh Mishra, Ingeniero consultor, Cisco Customer Experience

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).